2006

DIS-XML: Moving DIS to Open Data Exchange Standards

McGregor, Don

Paper 06S-SIW-132, Simulation Interoperability Standards Organization, 2006 Spring Simulation Interoperability Workshop, Huntsville, AL

DIS-XML: Moving DIS to Open Data Exchange Standards

Don McGregor
Don Brutzman
Curtis Blais
Adrian Armold
Naval Postgraduate School MOVES Institute
700 Dyer Road RM267
Monterey, CA 93943
mcgredo@nps.edu, brutzman@nps.edu, clblais@nps.edu, adarmold@nps.edu

Mark Falash
Eytan Pollak
L-3 Communications Link Simulation & Training (Link)
12351 Research Parkway
Orlando, FL 32826
mdfalash@link.com, epollak@link.com

Keywords:
DIS, XML, marshalling, unmarshalling, data binding

ABSTRACT: *The Distributed Interactive Simulation (DIS) IEEE-1278.1 standard is used in many military simulations. The binary data format for DIS does not lend itself to use in emerging web services standards or data analysis and storage tools because the data format is not recognized by most general purpose data manipulation tools. We examine an alternate XML-based representation of DIS information. An XML representation unlocks legacy data by placing it in a standard data format, and thereby integrates with the many XML data transformation, storage, and analysis tools. An XML-based approach also allows easy generation of programming language Application Program Interfaces (APIs). The information contained in DIS PDUs can shift between any of several representations, including the binary IEEE-1278.1 standard, XML, or programming language objects. The XML representation can be used in web services applications or passed over instant messaging channels. Initial tests demonstrate satisfactory performance.*

1. Overview and Motivation

Distributed Interactive Simulation (DIS) [1] is a communications protocol standard used in many military simulations, including large-scale High Level Architecture (HLA) federations through network bridges commonly called DIS gateways. The protocol defines, among other things, a set of Protocol Data Units (PDUs). The PDUs are specified in a binary format to promote efficiency. A single host may process thousands of PDUs per second, and binary formats are a more efficient format from which to extract information. The greater efficiency comes at a cost. Because the data is in a binary format it cannot be easily processed by tools that are unaware of the DIS standard. For example, it might be useful to search an archive of all PDUs exchanged in an exercise in order to find entities that passed through a particular area or that meet some other criteria. If the archived PDUs are in the binary DIS format this task is more difficult than it needs to be; the tools that search the PDU archive need to understand the IEEE standard format. It would be more useful to reuse powerful, industry standard tools to process the information, but these tools typically do not understand the binary DIS format.

Many DIS implementations read DIS PDUs from the network and transform them into programming language objects [2] [3]. Within the context of the simulation programming logic the PDUs are more easily handled as objects, typically with accessor methods to retrieve individual PDU fields. Users may modify the field values and then write the PDU back to the network in IEEE-1278.1 format. The process of changing between object and binary PDU format and back again is called marshalling and unmarshalling, or sometimes serialization and deserialization. The information contained in the two representations of the PDU is essentially identical, and differ only in format. This presents one possible avenue for tool-based processing of DIS information: the programming language objects. But this approach is ultimately unsatisfactory, for the same reason that the binary DIS format is problematic: the tools need to understand the programming Application Program Interface (API), and in general this is unlikely to be the case for third-party tools. What is needed is a format that can both represent the information in a PDU and be used by other tools, all without giving up the performance and backwards compatibility advantages of the IEEE-1278.1

binary format or the programming convenience of an object-based API. The Extensible Markup Language (XML) [4] is a natural choice to meet these requirements.

2. Background

Today's modern Command and Control (C2) and Modeling and Simulation (M&S) systems seek to exploit web-based standards and technologies to achieve the level of information and software access as is available in the Internet and World Wide Web. The Department of Defense is rapidly transitioning to a Global Information Grid (GIG) that will enable diverse systems across multiple communities of interest to readily exchange data and services to improve decision-making [5]. In 2002, the Defense Modeling and Simulation Office sponsored an initiative called the Extensible Modeling and Simulation Framework (XMSF) involving academia and industry (specifically, the Naval Postgraduate School, George Mason University, and Science Applications International Corporation, joined later by Old Dominion University's Virginia Modeling, Analysis, and Simulation Center) to promote application of web-based standards to military M&S. XMSF is "a composable set of standards, profiles and recommended practices for web-based modeling and simulation" [6]. Over the past 3.5 years, numerous research projects have been started to demonstrate application of XMSF principles, including:

- Extensible Battle Management Language (XBML): XML representation of Battle Management Language constructs for passing orders and commands to live, constructive, and robotic forces [7].
- XMSF Overlay Multicast (XOM): Demonstrates the ability to perform many-to-many multicast over an open network by employing middleware relay agents that cooperate to form an overlay "meta-network" that replicates the traditional multicast tree and achieves network use efficiency equal to Internet Protocol multicast [8].
- Web-Enabled Run-Time Interface (WE-RTI): Demonstrates ability to create an HLA federate that is able to communicate with a Run-Time Infrastructure (RTI) using Web Services [9].
- Scenario Authoring and Visualization for Advanced Graphical Environments (SAVAGE): Demonstration of tools, techniques, and capabilities related to the Extensible 3D Graphics (X3D) international standard for web-based 3D [10]
- Visual Simulation Toolkit (Viskit): Graphical User Interface for designing discrete event simulations using an underlying XML representation of the simulation event graph with auto-generation of Java code for execution [11].
- SAVAGE Modeling and Analysis Language (SMAL): XML language for describing objects and environment to initialize discrete event simulations.
- World-Class Modeling (WCM) Project: Event-based interactions via Web Services between the Naval Simulation System (NSS) and the Combined Arms Analysis Tool for the 21st Century (COMBAT^{XXI}). [12]
- XML-based Tactical Chat (XTC): Exploiting community open source XML-based chat mechanisms for human-to-human, human-to-machine, machine-to-human, and machine-to-machine interactions, including use as a transport mechanism for standards such as DIS PDUs [13].
- XMSF Profiles: Definition of formal technical specifications for application of interoperable web-based technologies to enable composable and reusable M&S [14, 15].
- Mediation Services: Web services for C2/M&S information exchange using the Command and Control Information Exchange Data Model (C2IEDM) [16].
- Mobility Common Operational Picture (M-COP): Specification of a common data model for representation and exchange of C2 and M&S information and processes relating to ground vehicle mobility data needed for maneuver planning [17].
- XML Schema-based Binary Compression (XSBC): Leveraging the structural information in XML Schema to provide efficient compression and decompression of XML data streams [18, 19].

It is evident from these examples that researchers are covering a broad spectrum of topics within XMSF, from policy and standardization issues, to modeling and networking frameworks, to simulation design and implementation, to cross-simulation (or C2 system) interactions, to efficient messaging and transport. The DIS-XML work described in this paper is another complementary effort under the XMSF umbrella. It extends the scope of application of the DIS standard to world-wide proportions through web-enabling the protocol. Details about the approach and capabilities are presented in the following sections.

3. DIS to XML

A fragment from the IEEE-1278.1 specification that specifies the binary layout of DIS is shown in Table 1.

The actual PDU has binary data at the offsets defined in the specification. Figure 1 shows what an XML representation of the data defined by the PDU fragment above might look like.

Field size (bits)	Entity State PDU fields	
96	PDU Header	Protocol Version – 8-bit enumeration
		Exercise ID – 8-bit unsigned integer
		PDU Type – 8-bit enumeration
		Protocol Family – 8-bit enumeration
		Timestamp – 32-bit unsigned integer
		Length – 16-bit unsigned integer
		Padding – 16 bits unused

Table 1. Binary Format DIS Specification Fragment

```
<PduHeader protocolVersion=”6”
           exerciseId=”0”
           pduType=”1”
           protocolFamily=”1”
           timestamp=”0”
           length=”144”/>
```

Figure 1. XML Equivalent of the Binary DIS Content

Once the data is represented in XML format, the full range of XML tools can be brought to bear on DIS data: archiving, transformation, database, web services, and so on. However, the XML format is obviously less efficient for hosts in high-traffic environments to process. What is needed is a way to shift the information contained in a PDU to the most convenient format for any particular context, so binary data can be used in high traffic environments, and XML for archiving and data exchange with systems outside of the traditional DIS problem space. Combined with the need for a programming language representation of the PDUs, there are at least three convenient ways to represent the data: IEEE-1278.1 binary, XML, and programming language objects. The representations of the data and transformations between the representations are shown in Figure 2.

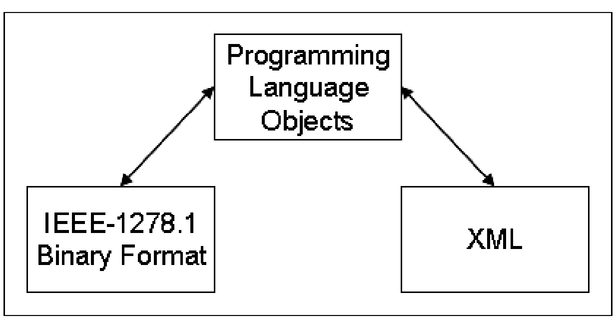

Figure 2. Translating DIS Data across Three Formats

The DIS data can of course be saved in still more formats—for example, a relational database. But there are many open source and commercial tools that allow users to map XML to other formats. XML is in effect a universal, platform-neutral format that allows the data to be easily transformed into any other format one may wish, usually with freely available tools.

4. Implementation

Our implementation of the ideas outlined above uses as its starting point an XML schema. Schemas are XML documents that describe the structure of an XML document. The World Wide Web Consortium specifies the format of schema documents [20]. We have written an XML schema that describes the major aspects of DIS PDUs. An example fragment from the schema is shown in Figure 3.

```
<xsd:complexType name="HeaderType">
     <xsd:attribute name="protocolVersion"
                    type="xsd:byte"
                    default="6"/>
     <xsd:attribute name="exerciseID"
                    type="xsd:unsignedByte"
                    default="0"/>
     …
     <xsd:attribute name="length"
```

Figure 3. XML Schema Fragment for DIS PDU Header Format

Some portions of the PDU header are omitted for brevity. As can be seen, the schema is in XML format, but describes the structure of another XML document, in this case one that contains a representation of DIS PDUs. The name and type of PDU header attributes are specified, along with any useful default values. A valid XML fragment as described by this schema fragment was shown in Figure 1. Writing the schema is a matter of reading the IEEE-1278.1 specification and writing a corresponding schema document that contains the same information.

Once the schema is written there are a number of tools that can automatically translate the XML schema into programming language objects. This is a major savings in programmer labor. We used Sun Microsystem’s Java Architecture for XML Binding (JAXB) [21]. JAXB takes the DIS XML schema as input and produces Java interfaces and classes that correspond to objects described in the schema. The generated classes have the ability to marshal and unmarshal themselves from XML documents, and have getter and setter methods for the attributes and values specified in the schema. An example code fragment from a generated Java language interface file is shown in Figure 4.

```
public interface HeaderType {
  byte getProtocolFamily();
  void setProtocolFamily(byte
value);
…
}
```

Figure 4. Java Language Interface File Generated from the DIS XML Schema

As mentioned, the generated Java classes can marshal and unmarshal themselves from XML documents. The example XML document in Figure 5 expresses a DIS PDU that can be directly read and written from the Java classes.

Note that not all values defined in IEEE-1278.1 are shown in the XML document. This is the result of the use of default values in the schema. If the value of the field does not differ from the default value, then it is not printed in the XML document. Its value is implicit, and retrieving the attribute or value from the API or XML tool will yield the default value.

The schema for DIS (which was written manually) and JAXB together gave us the ability to read and write XML to Java objects. The ability to marshal and unmarshal to IEEE-1278.1 is another matter. Obviously JAXB has no knowledge of DIS. The programmer must write code to marshal and unmarshal to binary format DIS. We have written code to do this for the major DIS PDUs. Similar code can be written or generated to marshal the PDUs to other formats or emerging standards, such as the Efficient XML Interchange (EXI) [22].

```
<DIS>
  <EntityStatePdu capabilities="0" entityAppearance="0" forceID="0"
    numberOfArticulationParameters="0">
    <PduHeader length="144" pduType="1" protocolFamily="1" timestamp="0"/>
    <EntityID application="1" entity="2" site="0"/>
    <Entity/>
    <AlternativeEntity/>
    <EntityLinearVelocity/>
    <EntityLocation x="62.0"/>
    <EntityOrientation/>
    <DeadReckoningParameters
       otherParameters="000000000000000000000000000000">
      <EntityLinearAcceleration/>
      <EntityAngularVelocity/>
    </DeadReckoningParameters>
    <EntityMarking characterSet="0" characters="0000000000000000000000"/>
  </EntityStatePdu>
</DIS>
```

Figure 5. Entity State PDU in XML Format

5. Example Uses

The NPS team and our industry partners have implemented several examples using DIS-XML.

X3D Scene Binding using Xj3D: X3D [23], from the Web3d Consortium (http://www.web3d.org), is an open standard for 3D content. Xj3D [24] is an implementation of an X3D browser. It includes an IEEE-1278.1 networking component, which can read selected DIS PDUs and use the PDU data to move objects in the 3D scene. As a simple test we devised a minimal 3D scene that could be driven by a Java application that generates binary DIS PDUs. The control panel and the Xj3D scene being driven by the controls are shown in Figure 6.

Figure 6. X3D scene in Xj3D With IEEE-1278.1 PDU Generator

DIS-XML across XMPP Chat. Extensible Messaging and Presence Protocol (XMPP) is an Internet Engineering Task Force (IETF) standard for instant messaging [25]. XMPP uses an open XML standard as a basis for conferences and instant messaging. The standard can be used for much more than just instant messaging; it is in effect an XML-enabled communications backbone that can be used to pass XML between automated endpoints. With the open source Wildfire XMPP server [26] and Smack API [27] we used XMPP as a multicast DIS bridge between networks that were not multicast-enabled. A process on one of the multicast islands reads IEEE-1278.1 binary PDUs, converts them into the XML representation, and sends them to an XMPP multi-user chat room (MUC). On the other network, a process reads XML from the XMPP MUC, unmarshals it to the Java object representation, and then marshals it out on the local network as IEEE-1278.1 binary format PDUs. For debugging purposes a user may optionally observe a text representation of the PDUs by simply joining the MUC with a standard XMPP client.

Figure 7 shows the earlier DIS-XML PDU example embedded in an XMPP message. Multiple DIS PDUs can be embedded in a single chat message, providing additional efficiency in bandwidth and processing.

```
<message id=”qoU4g-67”
  to=”adarmold@surfaris.cs.nps.navy.mil/Exodus
  type=”groupchat”
  from=”disxml@conference.surfaris.cs.nps.navy.mil/snerd0”>
  <body>A DISXML message from the sender. </body>
  <properties mlns=http://www.jivesoftware.com/xmlns/xmpp/properties”>
    <property>
      <name>disXML</name>
      <value type=”string”>
      <?xml version=”1.0” encoding=”UTF-8” standalone=”yes”?>
      <DIS>
        <EntityStatePdu capabilities="0" entityAppearance="0" forceID="0"
          numberOfArticulationParameters="0">
        <PduHeader length=”144” pduType="1" protocolFamily="1"
                  timestamp="0"/>
        <EntityID application=”1” entity=”2” site=”0”/>
        <Entity/>
        <AlternativeEntity/>
        <EntityLinearVelocity/>
        <EntityLocation x="62.0"/>
        <EntityOrientation/>
        <DeadReckoningParameters
          otherParameters="000000000000000000000000000000">
          <EntityLinearAcceleration/>
          <EntityAngularVelocity/>
        </DeadReckoningParameters>
        <EntityMarking characterSet="0" characters="0000000000000000000000"/>
        </EntityStatePdu>
      </DIS>
    </value>
   </property>
  </properties>
</message>
```

Figure 7. Entity State PDU in XML Format Embedded in XMPP Message

Performance is adequate for small quantities of PDUs when run on commodity laptop hardware. Approximately 100 PDUs per second can be sent across a local XMPP server. Further server performance optimizations are likely.

L-3 Communications Link Simulation & Training (Link): Internal Research and Development (IRAD) efforts have been pursuing commercial web-based and game-based technologies since 2004, including DIS-XML and Xj3D implementations. Link has produced numerous prototypes of a Next Generation Trainer (NGT) to demonstrate the feasibility of applying these technologies to the simulation-based training domain. During the 2004 Interservice/Industry Training, Simulation, and Education Conference, Link demonstrated a scenario comprised of an F16 flight simulator along with Aviation Combined Arms Tactical Trainer (AVCATT) Semi-Automated Forces (SAF) and Xj3D communicating via DIS-XML. This scenario was enhanced in early 2005 to a multi-site demonstration, which included four F16 simulators connected through a wide area network (L-3 Corporate Network). Two of the F16 simulators are located in Arlington, Texas with 2 more F16 simulators, AVCATT SAF, and Xj3D in Orlando, Florida. Exercise Control, Monitoring and Fault Insertion capabilities are located in Orlando, but controlled via the IOS in Binghamton, New York. Figure 8 illustrates where DIS-XML and Xj3D were applied in this configuration. Publishing and subscribing to data objects through the NGT Event Handlers provides the capability to communicate between simulation elements. Events are distributed using the XML Communications Bus and the Java DIS-XML gateway to native DIS applications by converting DIS-XML documents to DIS PDUs and vice versa. A total of approximately 30 entities were involved in the simulation, and real-time, man-in-the-loop performance was good, with no user-perceived latency in the display. Application elements interface to the Event Handlers using Java objects, which are based in part on the Java-DIS-XML implementation. The Event Handler converts the objects to DIS-XML documents and publishes them to subscribers. The XML communications bus was initially implemented using JBoss Java Messaging Service (JMS). There are ongoing efforts to evaluate alternative messaging and data communications mechanisms such as TCP multicast, SOAP, and HTTP variants.

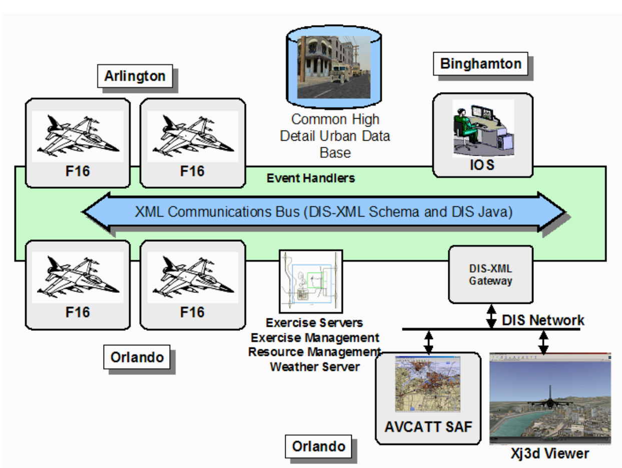

Figure 8. Link Four F16 Ownship Demonstration Configuration

Other Uses: The NPS Autonomous Unmanned Vehicle Workbench (AUVW) project [28] allows mission planning and analysis for unmanned submersibles and air vehicles. The AUV Workbench uses realistic physics models of undersea and air vehicles to generate DIS packets, which are used to visualize scenes in the Xj3D browser or embedded Xj3D application panel. The Xj3D browser has integrated DIS-XML as an alternative protocol stack for driving entities in the scenegraph. Entities in Xj3D can be configured to listen on XMPP multi-user chat rooms for DIS messages in XML format. Likewise, the AUV Workbench has been configured to allow sending DIS messages in XML format across the XMPP backbone.

MATLAB [29] is a scientific analysis tool that is widely used to analyze and chart data. The DIS-XML Java libraries can be called from within MATLAB. This allows MATLAB to read DIS network traffic or archived DIS PDUs from disk in either binary or XML format and make use of MATLAB's analysis facilities.

6. Future Work

XML's main benefit is that it is ubiquitous and often used as a standard for interchange between formats. While useful, this is really only interoperability on the level of syntax. Much more could be done if there was agreement at the level of data semantics. This requires agreement on the meaning of the XML elements and attributes. We have written an XML schema that describes the DIS PDUs. Another, equally valid schema written by another author may use different element or attribute names. The same semantic information is denoted by two different names, and exchanging XML data between applications using the two schemas will require manual disambiguation or other mapping operations. Just as existing applications can exploit commonality due to the standardization of IEEE-1278.1 binary PDUs, future applications can benefit from a single, agreed-upon standard for XML representations of DIS. The NPS DIS-XML schema and codebase has been submitted to the SISO DIS Working Gropu for further study and potential standardization.

XML, while a standard for interchange of data, imposes a significant load on hosts when it is processed. One idea that is being investigated by several companies and organizations is a binary format for XML. This would be more compact than text format XML and be much faster to parse, while remaining completely interchangeable with the original text XML. Standardization efforts are underway; the W3C has set up an Efficient XML Interchange (EXI) Working Group [22]. In the meantime, several companies are working on their own implementations, such as Sun Microsystem's Fast Infoset [30] and the MOVES Institute's Extensible Schema Based Compression (XSBC) [18]. A standardized binary XML format, would, when passed over the network, be somewhat larger than a hand-coded binary protocol, but probably easier and nearly as fast to parse. If DIS data is passed between applications in EXI format we anticipate that this will significantly increase the observed performance while retaining the benefits of XML. To further inform ongoing efforts, follow-on work will create a table of all DIS PDUs with size in bytes for DIS native binary format, DIS-XML text format, DIS-XML compressed under XSBC, and DIS-XML under the Fast Infoset.

7. References

[1] IEEE 1278.1a-1998, Standard for Distributed Interactive Simulation – Application Protocols.

[2] DIS for Java open source project site: http://sourceforge.net/projects/disjava

[3] MÄK Technologies VR-Link®, http://www.mak.com/vrlink.php

[4] World-Wide Web Consortium (W3C): Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation, 4 February 2004. http://www.w3.org/TR/2004/REC-xml-20040204

[5] Department of Defense: "Defense Acquisition Guidebook,"December 2004. Available at: http://akss.dau.mil/dag/Guidebook/Common_Interim Guidebook.asp

[6] C. Blais, D. Brutzman, D. Drake, D. Moen, K. Morse, M. Pullen, and A. Tolk: Extensible Modeling and Simulation Framework (XMSF) 2004 Project Summary Report, NPS-MV-05-002, Naval Postgraduate School, Monterey, CA, 28 February 2005.

[7] M. R. Hieb: "Extensible Battle Management Language (XBML): A Methodology for Web Enabling Command and Control for Network Centric Warfare," Proceedings of the 2004 Command and Control Research and Technology Symposium: The Power of Information Age Concepts and Technologies, San Diego, CA, June 2004.

[8] D. M. Moen and J. M. Pullen: "Performance Evaluation of the XMSF Overlay Multicast Prototype," Paper 05S-SIW-024, Spring Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, San Diego, CA, April 2005.

[9] J. M. Pullen, R. Brunton, D. Brutzman, D. Drake, M. Hieb, K. L. Morse, and A. Tolk: "Using Web Services to Integrate Heterogeneous Simulations in a Grid Environment," Proceedings of the International Conference on Computational Science, Krakow, Poland, 2004.

[10] D. P. Brutzman: Web-based 3D Graphics rendering of Dynamic Deformation Structures in Large-Scale Distributed Simulations, Naval Postgraduate School, Monterey, CA, 30 November 2003.

[11] A. Buss: "Simkit Analysis Workbench for Rapid Construction of Modeling and Simulation Components," Paper 04F-SIW-020, Fall Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Orlando, FL, September 2004.

[12] A. Buss and J. Ruck: "Using XMSF Web Services for Joint Modeling and Analysis," Paper 04F-SIW-021, Fall Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Orlando, FL, September 2004.

[13] D. Brutzman, D. McGregor, D. A. DeVos and C. S. Lee: XML-based Tactical Chat (XTC): requirements, Capabilities and Preliminary Progress, Naval Postgraduate School, Monterey, C A, 30 November 2003.

[14] K. L. Morse: "XMSF Profile Study Group Final Report," Paper 05F-SIW-013, Fall Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Orlando, FL, September 2005.

[15] R. P. Z. Brunton: "Documenting the Web Enabled RTI – An XMSF Profile Prototype," Paper 05F-SIW-093, Fall Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Orlando, FL, September 2005.

[16] B. Sun and A. Tolk: "C2IEDM Data Mediation Service," Paper 05S-SIW-120, Fall Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Orlando, FL, September 2005.

[17] N. C. Goerger, C. L. Blais, B. Gates, J. A. Nagle, and R. R. Keeter: "Toward Establishing the Mobility Common Operational Picture: Needs Analysis and Ontology Development in Support of Interoperability," Paper 06S-SIW-044, Spring Simulation Interoperability Workshop, Simulation Interoperability Standards Organization, Huntsville, AL, April 2006.

[18] T. D. Norbraten: Utilization of Forward Error Correction (FEC) Techniques with Extensible Markup Language (XML) Schema-Based Binary Compression (XSBC) Technology, Master's Thesis, Modeling, Virtual Environments, and Simulation (MOVES), Naval Postgraduate School, Monterey, CA, December 2004.

[19] M. E. Bayer: Analysis of Binary XML Suitability for NATO Tactical Messaging, Master's Thesis, Systems Technology, Naval Postgraduate School, Monterey, CA, September 2005.

[20] World-Wide Web Consortium (W3C): XML Schema Part 0: Primer Second Edition, W3C Recommendation, 28 October 2004. http://www.w3.org/TR/xmlschema-0
[21] Sun Developer Network: Java Architecture for XML Binding (JAXB). http://java.sun.com/webservices/jaxb/index.jsp
[22] World-Wide Web Consortium (W3C): Efficient XML Interchange Working Group Public Page. http://www.w3.org/XML/EXI
[23] ISO/IEC 19775:2004, International Standard: Information Technology – Computer Graphics and Image Processing – Extensible 3D (X3D) http://www.web3d.org/x3d/specifications/ISO-IEC-19775-X3DAbstractSpecification
[24] Web3D Consortium: Xj3D Working Group. http://www.xj3d.org
[25] Internet Engineering Task Force: Extensible Messaging and Presence Protocol (XMPP): Core, Network working Group Standards Track, P. Saint-Andre, Ed., October 2004. http://www.ietf.org/rfc/rfc3920.txt
[26] jivesoftware.org, Wildfire Server: the open alternative to proprietary enterprise instant messaging. http://www.jivesoftware.org/wildfire
[27] jivesoftware.org, Smack API: Simple and Powerful Java client API for XMPP. http://www.jivesoftware.org/smack
[28] Jeffery Weekley, Don Brutzman, Anthony Healy, Duane Davis, and Daryl Lee: "AUV Workbench: Integrated 3D for Interoperable Mission Rehearsal, Reality and Replay," Proceedings of the Mine Warfare Association Australian-American Mine Warfare Conference, Canberra, Australia, February 2004.
[29] The Mathworks, MATLAB. http://www.mathworks.com
[30] P. Sandoz, A. Triglia, and S. Pericas-Geertsen: Fast Infoset, Sun Developer Network, June 2004. http://java.sun.com/developer/technicalArticles/xml/fastinfoset

Acknowledgements

Thanks to Alan Hudson and Justin Couch of Yumetech for supporting implementation of DIS communications in the X3D specification and the Xj3D reference implementation of the X3D standard. Thanks to numerous NPS students who developed XML encodings of the DIS PDU formats. Thanks to the DIS Working Group for review and comment on early drafts of the paper. Thanks to the XBC Working Group for advancing requirements for binary XML encodings.

Author Biographies

DON MCGREGOR is a Research Associate in the Naval Postgraduate School's Modeling, Virtual Environments, and Simulation (MOVES) Institute. His research interests include binary XML, simulation, and high performance cluster computing. Mr. McGregor has an MS in Industrial Engineering from Oregon State University.

DON BRUTZMAN is a computer scientist and Associate Professor working in the Modeling, Virtual Environments, and Simulation (MOVES) Institute and Information Sciences Department at the Naval Postgraduate School. He chairs the Undersea Warfare Academic Committee and USW curriculum. His research interests include underwater robotics, real-time 3D computer graphics, artificial intelligence and high-performance networking. He is a member of the Institute of Electrical and Electronic Engineers (IEEE), the Association for Computing Machinery (ACM) Special Interest Group on Graphics (SIGGRAPH) and the American Association for Artificial Intelligence (AAAI). He serves on the board on the newly formed World Wide Consortium for the Global Information Grid (W2COG). He is a board member of the non-profit Camp SEA Lab Monterey Bay, which is building a youth-oriented year-round residential science camp. He is a founding member of the non-profit Web3D Consortium Board of Directors. He organized and led the SIGGRAPH Online 2001 team, designing and developing a huge audio/video/Web/XML information archive. Currently he leads the Extensible 3D (X3D) Working Group for the ISO X3D/VRML Specification, and directs development of numerous research projects as part of the Extensible Modeling and Simulation Framework (XMSF). Dr. Brutzman received a Ph.D. in Computer Science from the Naval Postgraduate School.

ADRIAN ARMOLD is a graduate student at the Naval Postgraduate School's majoring in Modeling, Virtual Environments, and Simulation (MOVES). His research interests include open source implementations and standards supporting chat mechanisms for tactical operations.

CURTIS BLAIS is a Research Associate in the Naval Postgraduate School's Modeling, Virtual Environments, and Simulation (MOVES) Institute. His research interests include multi-agent systems and operational ontologies for semantic interoperability. Mr. Blais has a MS in Mathematics from the University of Notre Dame. He is currently a Ph.D. candidate in MOVES researching an ontological framework for obtaining valued information at the right time in semantically rich network-centric architectures.

MARK FALASH is a Senior Principal Investigator for L-3 Communications Link Simulation & Training (Link) Internal Research and Development in Orlando, Florida. His current responsibilities involve investigation of technologies and products applicable to the next generation of Link training systems. He is the technical lead for projects applying web based, Information (IT) and Game Technologies to training systems, development of training preparedness and readiness tools targeting homeland security opportunities, and core architecture and infrastructure for a product line of virtual training simulators and reconfigurable simulators. He has published several papers on common architecture approaches for virtual training systems, IT and web based infrastructure and use of Operational Analysis in a training system for Homeland Security. He received an M.S. in Computer Science from California State University, Chico.

EYTAN POLLAK is currently director of Strategic Technologies at L-3 Communications Link Simulation & Training in Orlando, Florida. Recent projects include the development of operational analysis techniques, common simulation architectures, reconfigurable simulators, and synthetic environment technologies. Dr. Pollak has provided technical leadership and guidance for distributed training systems applied to aviation and ground platforms and external research programs. He has published papers and awarded patents on operational training analysis, common architectures for training systems, embedded simulation, and reconfigurable simulators. Dr. Pollak serves as adjunct professor of Electrical Engineering at the University of Central Florida. He received his Ph.D. from Purdue University.